Monitoring the Impact of Norms upon Organisational Performance: a Simulation Approach

Chris Haynes, Simon Miles and Michael Luck

Department of Informatics
King's College London, United Kingdom
christopher.haynes@kcl.ac.uk

Abstract. Normative organisations use norms to guide and constrain agent behaviour in order to facilitate cooperation. Norms and their associated enforcement strategies are chosen to further organisational goals, but the effect that a norm has upon organisational performance may change over time in a dynamic environment and behaviour that is desirable in one environment may come to be harmful in another. In this paper we seek to answer the question — how can an organisation detect when a norm is no longer supporting its goals? Specifically, how can it monitor the impact of a norm upon its performance? This paper has three contributions: first, we detail a model which relates an organisation's norms to its performance, second, we propose a mechanism for monitoring the impact of a norm upon that performance using a simulation approach, and finally we describe an implementation of our mechanism.

Keywords: norms, normative organizations, organisational performance, monitoring, simulation

1 Introduction

In organisations, agents work together to achieve goals that are difficult or impossible to achieve by a single agent alone. Cooperating agents within an organisation can solve problems with distributed resources, data or expertise, as well as long term problems in timescales beyond the life of an individual [5]. However, this cooperation can be complex to orchestrate, especially as the number of agents increases. In particular, agents may interfere with one another while performing actions, or come into conflict over shared resources.

Norms have been proposed as a solution to these coordination problems [9], and constrain agent behaviour by prohibiting, or obligating, certain actions or states of affairs. Autonomous agents may violate norms in exceptional circumstances, when the benefits outweigh the costs of punishment, so a normative organisation may be more flexible and robust in the face of unexpected problems, such as environmental changes. Conflicts may thus be reduced and goal achievement facilitated without rigid rules. This follows the example of the use of norms within human organisations as a means of management control over employee behaviour in order to improve organisational performance [22]. However, the design of norms is complex and challenging [21] with

behavioural controls leading to unintended consequences (as Merchant notes in human organisations [16]). Offline design of norms allows an organisation to select its initial norms [21], but if the environment is dynamic or largely unknown at design time, norms may come to have different practical effects from their designed purpose, so that the *impact* of the norms may change. This effect has been noted and studied in human organisations [23].

The problem of norms becoming counter-productive is considered by Boella et al. [2], who take the view that each norm is designed with an intended goal in mind, much like a plan, and that in each situation the norm must be interpreted to determine if it furthers that goal. They propose a logical framework to allow such interpretation, either by an agent choosing whether to comply with a norm, or by an enforcement mechanism deciding whether to sanction a violation. This interpretation makes use of constitutive norms that specify counts-as rules for variables that make up the regulative norms. While the approach is a valuable way to ensure that the effect of norms remains true to their intended purpose, it relies on agents (or the enforcement mechanism) having the capability and knowledge to perform such reasoning. Moreover, it does not concern itself with the situation where changing circumstances render the intended purpose of the norm itself harmful to the organisational performance. It is this latter problem that we seek to address in this paper — how can an organisation detect when a norm is no longer supporting its wider goals? More specifically, how can it monitor the changing impact of a norm over time in a dynamic environment?

This paper thus has three contributions. First, we propose a model of *norm impact* that links organisational norms to organisational performance. Second, we detail a mechanism to monitor norm impact within an organisation. Finally, we implement our mechanism and show how it reveals a change in impact in a dynamic environment.

As a motivating example, we use a mobile distributed sensor network (MDSN): specifically, a group of unmanned aerial vehicles (UAVs) undertaking a search and rescue task in which they seek to locate victims and report their locations to a control centre [11], described in Section 2. The model of organisational performance and norm impact is then detailed in Section 3, and the monitoring mechanism is described in Section 4, where we consider how to determine what is required in order to monitor norm impact within an organisation, and propose a method to do so. In Section 5 we describe an implementation of the norm impact monitoring system and present results. We finish with a review of related work in Section 6, and conclusions in Section 7.

2 The Scenario: UAV Search and Rescue

In order to motivate and illustrate our work, we adopt a search and rescue scenario involving an organisation using a team of UAVs to search for hidden victims. The organisation consists of a static *controller* and five mobile agents (representing the UAVs) who play the roles of *team leader* and *sensors*. The environment is represented by a grid of 50x50 cells in which some are open ground while others contain either bushes or trees. The *sensors* traverse the environment searching for victims, but their range of detection is limited to the cell they occupy and to adjacent cells.

If a victim is found, a *sensor* sends its location to the *team leader* and the information is relayed to the *controller* which keeps count of located victims. Another victim then appears at a random location, so that at any time there are always 6 hidden victims. Organisational performance is measured by the cumulative number of victims located.

A single norm ($avoidTrees$) prohibits a UAV from flying below 50 metres in a cell containing trees. As well as punishment for norm violation, flying low over trees gives a small risk of crashing, resulting in an agent being disabled for 80 time steps. If agents comply with the norm, they will not crash. The probability of a *sensor* detecting a victim depends on the contents of the cell (open, bush or tree) and the altitude of the sensor (flying low increases the probability). This is high in open ground, lower in bushes and even lower in trees. Over trees there is thus a tension between flying low to increase the chance of detecting a victim and flying high to reduce the chance of crashing.

The probability of crashing depends on conditions that can change over time, so it is possible that the usefulness of the norm to the organisation may also change. For example, on a calm, clear day, a UAV is able to fly at a low altitude over trees with barely any chance of crashing, and in such an environment the norm may harm performance by obliging it to fly high unnecessarily. In contrast, on a windy day, flying low makes a crash very likely, and the norm may greatly benefit performance. Note that in both cases, the goal of the norm (to stop UAVs crashing) has not changed, merely its impact. It is this change in norm impact that we seek to detect.

3 A Model of Organisational Performance

The term *organisation* encompasses many different types of multi-agent groupings, from hierarchies for specific tasks to loose congregations with common interests [13]. There are several well-established organisational models in the literature (e.g. MOISE [12]) but, in order to maintain generality as far as possible, we do not use an existing one here. However, since we are concerned with the impact of norms upon organisational performance, we do have some requirements. In particular, we are concerned with organisations with three characteristics: they have explicit, measurable goals; they comprise agents playing roles, each with defined responsibilities and tasks, so as to achieve those goals; and they use norms as a flexible means to guide and constrain behaviour.

With these characteristics in mind we represent an organisation as a tuple:
$O = \langle G, R, N, A, Mem, rwd \rangle$, where G is the set of goals, R is the set of roles, N is the set of organisational norms, A is the set of agents within the organisation, *Mem* is the organisational memory and rwd is the reward function that the organisation uses to measure its success at achieving its goals. We define these elements in the following sections.

3.1 Organisational Goals and Roles

Organisations can be seen as goal-directed systems designed to solve problems that are too large (either spatially or temporally) or too complex to be solved by individual agents [5]. In this work, we specifically concern ourselves with organisations that have a set of explicit goals, G, such that success at meeting those goals can be measured by

a reward function, rwd. A goal in this context is a state of affairs that the organisation wishes to achieve or maintain. This requirement is driven by our need to measure the performance of the organisation, which we consider in Section 3.5.

A *role* can be seen as a building block of an organisation, representing some service or function that must be performed by it, and related to other roles by interaction and authority relationships [8]. Designing organisations using roles allows the designer to abstract away from specific agents and instead impose goals and norms onto roles rather than individuals. During the operation of the organisation, these roles are filled by agents who seek to achieve the goals under the constraints of applicable norms.

3.2 Organisational Norms

We take norms to be obligations, prohibitions and permissions that constrain and guide behaviour [15, 1]. We consider prohibitions as negative obligations: one is obliged to see to it that the prohibited action or state does not occur. In our model, a norm is represented[1] as: $n = \langle target, type, content, context, punishment \rangle$ where $target$ is the role to which the norm applies, $type \in \{obl, per\}$ is the deontic type of the norm (either obligation or permission), $content$ is the action or state referred to by the norm, $context$ is a statement of under which circumstances the norm applies, and $punishment \in \mathbb{R}^+$ is the penalty for violating the norm. Permissions cannot be violated, so have no associated punishment. Note that this representation does not consider norms where the punishment is the imposition of a contrary-to-duty norm. Both $content$ and $context$ are boolean expressions that can be understood by agents playing the roles, and can thus use environmental variables, internal agent variables, or actions. For example, a norm, $avoidTrees$, which prohibits an agent playing a $sensor$ role from flying below fifty metres in altitude over trees is represented in the following way: $avoidTrees = \langle sensor, obl, altitude > 50, isOverTrees, 35 \rangle$.

Norms apply to any agent playing the targeted role and, if agents play multiple roles, the norms from all the roles apply. The organisational norm set, $N = \{n_1, n_2, ..., n_i\}$, contains the norms used by the organisation. $N_{r_i} \subset N$, is the set of norms that apply to role r_i (and hence any agents playing that role). In our UAV scenario, there is a single norm, $avoidTrees$, applying to the $sensor$ role, so $N = N_{sensor} = \{avoidTrees\}$.

3.3 Agents

Agents play the roles within an organisation and it is the aggregate actions of these agents that make up the organisation activity. An agent playing a role in the organisation commits to try to achieve certain goals at certain times. These goals may be sub-goals of larger missions performed by multiple agents. We assume that agents are benevolent towards their organisation, and that they will attempt to achieve their goals. However, since agents are autonomous, they have leeway in *how* they achieve the goals. They may also have multiple goals at any one time and so must decide the order in which to

[1] Our representation is a simpler version of Andrighetto's [1]; we do not include norm defender or source since norms are imposed by the organisation and self-enforced by agents themselves.

pursue those goals. In this section, we consider how agents within normative organisations choose which actions to take. We specifically examine BDI agents using offline planning.

BDI agents [4], such as those using AgentSpeak(L) [19], perform offline planning using libraries of pre-compiled plans to achieve goals and react to events. To enable flexible plan selection in the presence of norms, we follow the approach of Oren et al. [18], where each agent has some utility function that it seeks to maximise and in this respect, achieving goals increases its utility, while violating norms decreases its utility. There is no requirement for an agent's utility function to directly relate to the organisation's reward function (described in Section3.1), in particular the agent may lack the wider knowledge required to determine whether its actions are ultimately beneficial to the organisational goals. Now, since agents are benevolent towards the organisation, they automatically reduce their utility for norm violations, so in effect punish themselves rather than rely on external enforcement — they do not wish to evade the consequences of their violations.

Given this, we can model an agent, a as $a = \langle G_a, \Pi_a, B_a, R_a, C_a, PlanSelection_a \rangle$ where G_a is the set of current goals, Π_a is the set of available plans, B_a is the set of agent's beliefs (including its representation of the environment), R_a is the set of roles currently played by the agent. C_a is a set of capabilities $\{c_1, c_2, ..., c_n\}, c_i = \langle \alpha, comp \rangle$, where α is some action that the agent can perform and $comp \in [0, 1]$ represents the agent's level of competence at performing the action. The set of plans, $\Pi_a = \{\pi_1, ..., \pi_n\}$ represent the contents of the agent's *plan library*. $PlanSelection_a$ is the set of two functions, $selectPossiblePlans_a$ and $selectBestPlan_a$ that the agent uses to select a plan from the plan library when deciding upon a course of action.

An agent chooses its plan based upon its current goals, beliefs, capabilities and norms. Since violating a norm reduces utility, agents prefer to comply with applicable norms, but if a goal is very important and yields a high utility, then it may make sense for an agent to violate a norm in order to achieve it.

In order to choose a plan, first the agent generates a set of possible plans, Π_{pos}, using $selectPossiblePlans_a(G_a, \Pi_a, B_a, C_a) \rightarrow \Pi_{ap}$. Then it selects the plan yielding the highest utility, π_{N_a}, using $selectBestPlan_a(\Pi_{pos}, B_a, C_a, N_a) \rightarrow \pi_{N_a}$, where N_a is the set of norms applicable to the agent. Using this method of plan selection, if all else is equal, the selected plan, and hence the agent's behaviour, is dependent upon the norms.

3.4 The Environment and Agent Activity

An organisation acts within and upon an environment in order to achieve its goals, and this activity may change the environment. We must therefore represent this environment, and also the activity itself. We model the environment as a set of variables, with each variable referring to a property of the world. At a specific time t_i, the environment is E_{t_i}, where $e \in E_{t_i}$ is an environmental property.

The *activity* of an organisation is the aggregate actions of all the member agents. AC_{org,N,t_i} is the organisational activity at time t_i and is defined as

$$AC_{org,N,t_i} = \bigcup_{j=0}^{|A|} AC_{a_j,N,t_i}$$

where A is the set of all agents in the organisation, and AC_{a_j,N,t_i} is the set of actions taken by agent a_j, at time t_i, with organisational norm set N.

To represent the changes that an organisation's behaviour makes upon the environment we specify an environment transformation function, $\mathcal{ETF}(E_{N,t_i}, AC_{org,N,t_i}) \rightarrow E_{N,t_{i+1}}$, where AC_{org,N,t_i} is the *organisational activity* at time t_i using organisational norm set N, and $E_{N,t_{i+1}}$ is the environmental state arrived at after one 'tick' of activity by the organisation. Note that the environment may change without the involvement of agents.

Over a time period longer than a single 'tick' (from t_0 to t_n), if the norm set N does not change, we can define an extended transformation function:

$$\mathcal{ETF}_{t_0 \rightarrow t_n} \left(E_{N,t_0}, \bigcup_{i=0}^{n} AC_{org,N,t_i} \right) \rightarrow E_{N,t_n}$$

where E_{N,t_0} is the environmental state at t_0, and E_{N,t_n} is the state at t_n.

3.5 Organisational Performance and Norm Impact

We are interested in how the norms of an organisation help it to perform its tasks given a specific environment and set of agents playing its roles. We informally define organisational performance over a time period as the increase in organisational utility over that period; this may be positive (if utility increases), negative (if it decreases) or zero. In this section, we consider how an organisation may calculate this performance, and propose a definition for norm impact.

An organisation's reward function may not depend purely on the current environmental state, but may also depend on past states and organisational behaviour. To accommodate this we use the concept of *organisational memory*, a repository of information that an organisation updates over time (though simple organisations may not require such a memory). We denote organisational memory at time t_i as Mem_{N,t_i}, where N is the organisational norm set (which has not changed from t_0 to t_i), and $m \in Mem_{N,t_i}$ is some variable derived from environmental, organisational or agent properties over time.

The reward function is defined as $rwd(E_{N,t_i}, Mem_{N,t_i}) \rightarrow utility_{N,t_i}$, where $utility_{N,t_i} \in \mathbb{R}$ is the organisational utility, and E_{N,t_i} is the state of the environment at time t_i resulting from the activity of the organisation using norm set N (see Section 3.4). $utility_{N,t_i}$ indicates how well the organisation is performing at time t_i using norm set N. In our UAV scenario, the number of victims found measures the performance of the organisation over the mission: a higher number indicates better performance.

We define organisational performance over a time period t_s to t_f as the difference between the utility values at those times (so long as N does not change over the period):

$$\mathit{perf}_{N,t_s \rightarrow t_f} = utility_{N,t_f} - utility_{N,t_s}$$

Thus performance is the measure of change in the organisation's utility from t_s to t_f.

We define *norm impact* as the effect of a specific norm upon organisational performance. Formally, the impact of norm n between time t_s and t_f is

$$impact_{n,t_s \rightarrow t_f} = \mathit{perf}_{N,t_s \rightarrow t_f} - \mathit{perf}_{N',t_s \rightarrow t_f}$$

where $N' = N \setminus \{n\}$, and $perf_{N',t_s \to t_f}$ is the performance that would have occurred if the organisation had been using norm set N' over the time period. The impact of a set of norms can also be derived in a similar way.

The challenge is to derive the performance for both norm sets, N and N', over the same time period, since only one norm set would actually be applied within the organisation at any one time. If N is the applicable norm set, the performance under N' must be estimated. In the remainder of this paper, we describe our proposal to estimate this performance using a simulation approach.

4 Monitoring Norm Impact

In this section we propose a method to monitor the impact of a norm, n, upon the performance of an agent organisation. We do not measure norm impact directly, but, instead, use a simulation approach where the activity of the organisation is modelled twice: first, using the existing set of organisational norms, N, and then under the norm set, N', where $N' = N/n$. Each simulation starts from the same state derived from the state of the organisation and environment at a specific time, but with different norms. First, we specify the process and provide a monitoring algorithm, then we examine the process in more detail.

4.1 The Monitoring Process

The monitoring entity may be an agent (for example, one playing a monitoring or leadership role), or it may be some program either internal or external to the organisation. We make no assumptions about its nature, but refer to it as the *monitor*. The monitoring process has two stages. First, the *monitor* captures information about the organisation, agents and the environment at a single point in time — the *snapshot*, which is the state of the system at that moment in time (in Section 4.2 we examine what needs to be in this snapshot). The *monitor* uses the snapshot to build a model of the environment and the agents. In the second stage, the *monitor* uses these models to simulate agent activity over the time period required, both with and without the norm of interest. Multiple runs of each simulation may be needed to get an average performance if the environment is non-deterministic.

Algorithm 1 shows the monitoring process algorithm for a single time period, from time t_s to t_f, of $length$ 'ticks'. Norm set N' is generated by removing the investigated norm, n, from the organisational norm set, N (line 1). At t_s the *monitor* gathers the snapshots (lines 4 to 6). Using this information it creates two simulations, sim_N and $sim_{N'}$, differing only in the norm sets used (lines 7 and 8). The organisational performance under each set of norms, p_N and $p_{N'}$, is estimated by running the simulation for the required number of 'ticks' to obtain the environment state and organisational memory at time t_f, and then calculating the performance based on the organisational reward function (lines 9 to 12). Finally, norm impact, $impact_n$, is calculated by a simple subtraction (line 13).

We choose to create sim_N and generate p_N rather than compare $p_{N'}$ directly to the performance seen in reality for two reasons. First, in a non-deterministic environment,

Algorithm 1 The monitoring process, for a single monitoring period

Require: A multi-agent organisation to be monitored, *MAS*
Require: Its norm set N
Require: Its set of agents A
Require: Its organisational specification, Org
Require: The norm to be investigated n.
Require: A monitoring time period, $mt = \langle t_s, t_f, length \rangle$
Generate norm set $N' \leftarrow N/\{n\}$
while *MAS* is active **do**
  if time = t_s (the start of a monitoring period) **then**
    Gather snapshot of environment properties, E_{N,t_s}
    Gather snapshot of agent states, AS_{N,t_s}
    Gather snapshot of organisational memory, Mem_{N,t_s}
    Create simulation, $sim_N \leftarrow createSim(A, E_{N,t_s}, Mem_{N,t_s}, Org, N, AS_{N,ts})$
    Create simulation, $sim_{N'} \leftarrow createSim(A, E_{N,t_s}, Mem_{N,t_s}, Org, N', AS_{N,ts})$
    Run simulation, $E_{N,t_f}, Mem_{N,t_f} \leftarrow runSim(sim_N, length)$
    Run simulation, $E_{N',t_f}, Mem_{N',t_f} \leftarrow runSim(sim_{N'}, length)$
    Estimate performance, $p_N \leftarrow rwd(E_{N,t_f}, Mem_{N,t_f}) - rwd(E_{N,t_s}, Mem_{N,t_s})$
    Estimate performance, $p_{N'} \leftarrow rwd(E_{N',t_f}, Mem_{N',t_f}) - rwd(E_{N,t_s}, Mem_{N,t_s})$
    Calculate norm impact, $impact_{n,t_s \rightarrow t_f} \leftarrow p_N - p_{N'}$
  end if
end while

a single run of the system may produce a performance that is exceptionally good or bad due to random chance, and this may obscure the impact of the norms. Using an average performance derived from multiple runs reduces this effect. Second, by comparing the performance in the real system to p_N the organisation may be able to detect problems with the simulation models. For example, if p_N is very different from the real performance (beyond that which could be derived from random chance), it is possible that the models used to create the simulations are not fit for purpose.

The remainder of this section examines the steps of the process in more detail. In Section 4.2 we consider the snapshot information and discuss what must be included, then we discuss the creation and running of the simulations in Section 4.3.

4.2 The Snapshot Information

The snapshots form the initial state of the simulations, as they capture the moment in time from which the simulated activity begins. The *monitor* gathers this information from the multi-agent system (MAS) either directly, or via the organisational agents themselves. There are three sources of snapshot: the environment, the organisational memory and the agents. In our model, the environment snapshot at time t_s is E_{N,t_s}, the set of environment variables at time t_s. Similarly, the organisational snapshot is Mem_{N,t_s}, the organisational memory at time t_s. The snapshot of an agent is the state of that agent at a moment in time, so if the MAS and the simulation use the same software framework, it may be possible simply to copy the entire internal state of the agent from the MAS and then use it in the simulation. However, we cannot make this

item	description	source
$selectPossiblePlans_a$	function to select possible plans	Agent
$selectBestPlan_a$	function to select best plan	Agent
R_a	set of roles played by the agent	Agent
G_a	set of a's goals	Organisation
N'_a	set of norms applicable to a	Organisation
Π_a	set of plans in a's Plan library	Agent
B_a	set of a's beliefs	Agent
C_a	set of a's capabilities	Agent

Table 1: Information required to simulate agent a's goal and plan selection

assumption about the nature of the MAS and the simulation without overly constraining the generality of our work. Therefore, in this section, we consider which parts of an agent's internal state must be included in a snapshot in order to simulate its behaviour.

The *monitor* must simulate the behaviour of the organisational agents during the monitoring period. First, agents select a goal based on their role, their available plans, and the environment. Second, they select a plan from the set of those applicable, based upon the norm set and the environment. Third, they perform the actions from the plan until the task is complete or abandoned. In our model, we represent the agent's plan selection mechanism as two functions: *selectPossiblePlans* and *selectBestPlan* (see Section 3.3); these are the functions that the *monitor* must simulate. In order to do so under norm set N'_a, the *monitor* needs eight pieces of information as shown in Table 1.

Agents within an organisation have goals imposed upon them as a result of their acceptance of a role; these organisational goals take precedence over any individual agent goals. Role-based goals are available to the *monitor* via the organisation's specification, so the *monitor* can obtain G_a. It also has access to the organisational norm set and can thus derive N'_a, which are the norms applying to a without the norm of interest, n.

The *monitor* must rely upon the agents to provide the other information. However, in this paper we assume that most of these elements do not change over the lifetime of the MAS[2]. Specifically, we assume that the two selection functions ($selectPossiblePlans_a$ and $selectBestPlan_a$), the plan library, Π_a, the roles played by the agent, R_a, and its capabilities, C_a, do not change. The *monitor* therefore receives this information once when the MAS starts and uses the same information throughout its lifetime. This leaves the beliefs, B_a as information that must be provided in the agent snapshot.

If an agent is following one or more plans at time t_s, they must be recorded in the snapshot, because these plans may still be valid and sound under the new norm set. We should not require the agent to discard an existing plan purely because the norm set has changed. In the case of a long term plan this could severely harm organisational performance, though the effect could be minor and acceptable if the plans are short compared to the length of the time period of the simulation.

An agent may gather percepts from the environment (or other agents) but not process them immediately. For example, if an agent able to process a single message per

[2] If the agent playing a role changes, then these assumptions may be incorrect — we leave this for future work.

Algorithm 2 Starting an agent from a snapshot.

Require: Agent beliefs from snapshot B_{snap}
Require: Current plan from snapshot π_{snap}
Require: Agent goals G_a, and agent capabilities C_a
Require: Norm set used in simulation N_{sim}.
Require: Set of roles played by agent R_a
Require: Agent plan library Π_a
1: $B_a \leftarrow B_{snap}$
2: $N_a \leftarrow setNorms(R_a, N_{sim})$
3: $\pi_a \leftarrow \pi_{snap}$
4: **if** not $sound(\pi_a, I, B_a)$ **then**
5: $\quad \Pi_{pos} \leftarrow selectPossiblePlans(G_a, \Pi_a, B_a, C_a)$
6: $\quad \pi_{N_a} \leftarrow selectBestPlan(\Pi_{pos}, B_a, C_a, N_a)$
7: **end if**
8: **while** true **do**
9: $\quad$ Continue Agent Activity
10: **end while**

time step receives multiple messages, then it must store messages until able to process them. When taking a snapshot of such an agent, we must include those messages in our snapshot. Likewise, if an agent stores events for future processing, this event queue must be included, even though these events occurred before t_s. Percepts and messages processed by the agent prior to t_s need not be recorded in the snapshot because these will have been incorporated into the agent's beliefs and intentions if necessary.

4.3 Creating the Simulations

The direct effect of agent actions is one cause of environmental change, but there are other causes that must be considered. The actions of agents have a direct effect on the environment (or agents), but may also cause side-effects on either the environment or the agents. There may also be an element of environmental dynamism that is unrelated to the organisation's agents. The *monitor* must thus simulate three aspects: the behaviour of the agents and their direct effect on the environment; the indirect side-effects of those actions; and other elements of dynamism unrelated to agent actions.

In order to model indirect changes, the *monitor* requires a model of the environment to determine likely side-effects of agent actions upon the environment and also to simulate changes arising from environmental dynamism. This may include modelling physical effects (if the MAS exists within a physical environment). For example, in our UAV scenario the simulation must model the probability of a low-flying UAV crashing into a tree. Also, it must have a model of other agents sharing the environment with the organisation, so that their actions can be simulated.

Starting a simulation from a snapshot requires the agents to begin *in media res*, that is, as if they were in the middle of whatever situation they were in when the snapshot was taken. Algorithm 2 shows the required steps for an agent starting from a snapshot — beliefs and the current plan are derived from the snapshot, and the set of applicable norms are generated from the new organisational norm set, N_{sim} and the set of roles

Table 2: Scenario Parameters

(a) Detection probabilities

Cell Terrain	Probability (fly low)	(fly high)
open	1.0	0.7
bush	0.8	0.4
tree	0.5	0.1

(b) Plan properties

Plan	Properties success	reward	Weight
$searchHigh$	0.1	40	4.0
$searchLow$	0.5	5	2.5

played by the agent, R_a. The agent must determine whether the current plan is sound (possible and desirable), and if not then it reconsiders its course of action.

4.4 Calculating Performance and Norm Impact

Once the simulations (sim_N and $sim_{N'}$) have been created, the next step is to run them and calculate the organisational performance. As in Algorithm 1, each simulation is run over the time period of interest (t_s to t_f) and the simulated environmental state and organisational memory at the end of the run are used in the reward function to calculate organisational utility at time t_f under both norms sets, N and N'. If the environment is non-deterministic, then multiple runs may be performed and an average utility calculated for each norm set. Performance is then calculated by subtracting the starting utility (which is calculated from the snapshot information). Finally, the norm impact is calculated by subtracting the performance under N' from the performance under N.

5 An Implementation of the Monitoring System

To evaluate our monitoring mechanism, we implemented a monitoring system for our UAV scenario (described in Section 2). In this section we describe our implementation in a broad fashion, but for reasons of brevity and clarity have omitted much of the detail.

A MAS representing our UAV scenario was built using the Jason AgentSpeak framework [3]. It consists of a single *controller* and five *sensor* agents, one of the latter also plays the *team leader* role. The *sensors* have a library of plans allowing them to search the environment as a team, coordinated by the *controller* and *team leader*. The victims are also agents. As described above, the probability of detecting a victim located in a cell varies with terrain type and altitude, shown in Table 2a.

We do not implement a full normative plan selection mechanism as described by Oren et al. [18], but instead the *sensors* have two plans for searching a tree-filled cell, one compliant (*searchHigh*) and one that violated the norm (*searchLow*). The plans are annotated with two properties, *success* and *reward*, to reflect the probability of successfully searching a cell, the reward for doing so, and the reward reduction for violating a norm. The reward is set to a value of 40 and the punishment for violating the norm is set to 35. The *success* property is determined by the values in Table 2a.

The plan selection function, *selectBestPlan* uses weights derived by multiplying the plan properties, *success* and *reward* to give an expected reward for following each

Table 3: Environmental properties

Environment	Crash probability	Delay time
Low Crash	0.005	80
Mid Crash	0.05	80
High Crash	0.1	80

plan (see Table 2b). The agents then choose the plan with the highest weight. Given these weightings, agents choose to comply with the norm since the expected utility gain is higher from the compliant plan. The organisational performance over a time period is measured by the number of victims located during that period. For example, if at time t_1, 6 victims have been found, and at time t_2, 11 victims have been found, the organisational performance is 5. This count of located victims is performed by the *controller*, so we do not use an explicit organisational memory.

In order to implement Algorithm 1, we first need to create snapshots of the environment and agents. The environment snapshot is a file generated by Jason that records the environmental state at the start of the monitoring period. To generate the agent snapshots, we use a new Jason internal action[3], *dumpState*, to record an agent's beliefs and message queue in a file. This internal action is performed as part of a plan triggered at the start of each monitoring period.

The simulations were also built using Jason, with the environment and the agents reading in the snapshot information at the start of the simulation. A new internal action, *bootstrap*, allows an agent to copy the beliefs and message queue from the snapshot, and a plan is triggered on start-up to restart agent activity based on beliefs (specifically, the point reached in traversal of the area, and whether currently disabled due to a crash). One simulation uses the $avoidTrees$ norm, the other does not — implemented by changing the *reward* property for the *searchLow* to 40 (to represent the absence of a penalty). Each simulation is run 100 times, and the organisational performance calculated from the number of victims found over the period.

5.1 Experiments and results

The MAS ran for 3000 time steps and norm impact was monitored over periods of 500 time steps, beginning respectively at time 500, 1500 and 2500. We performed experiments to represent three environments differing in the probability of a UAV crashing if flying too low over a tree: a high crash, low crash, and varying environment. The consequences of a crash were the same in all environments; Table 3 details the values of these properties. The varying environment began as a high crash one until time 1251, then became a mid crash environment until time 2251 when it became a low crash one.

Snapshots of the system were taken at time 500, 1500 and 2500, and used as the basis of simulated activity both with and without the norm, $avoidTrees$. Norm impact over the monitoring period was then calculated, with results displayed in Figure 1. The norm had a positive impact in the high crash environment and a negative impact in the

[3] Jason internal actions are Java functions that allow agents to perform actions not related to their environment, such as writing to a log file.

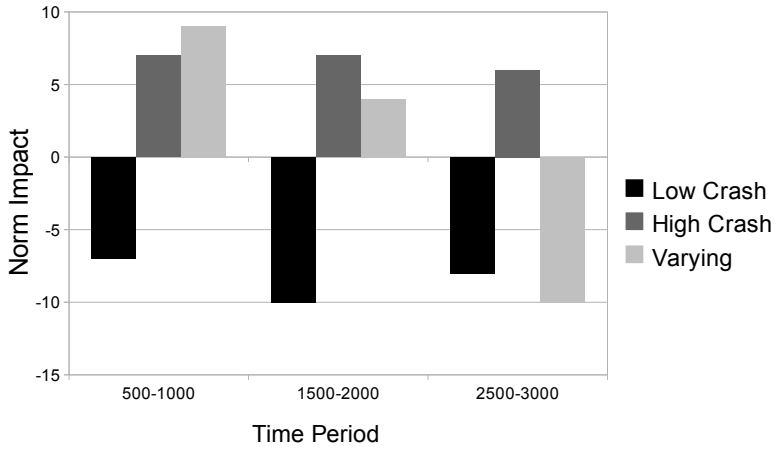

Fig. 1: Impact of norm $avoidTrees$ in different environments over three time periods

low crash environment. In the varying environment, norm impact begins positive, but changed to become negative as the probability of crashing reduces. These results show that it is possible to quantitatively measure norm impact and to measure the change of this impact as the environment changes. An organisation using such a monitoring mechanism is thus able to detect when a norm is becoming less useful, or indeed harmful, to its overall goals.

6 Related Work and Conclusions

The two main approaches to the design of norms are offline design [21] (where norms are designed prior to execution), or an emergent approach where satisfactory norms are converged upon by agents through a learning process [20]. Both have problems for situations in which an organisation wishes to use norms to regulate behaviour in a dynamic environment. With an offline approach, norms are designed for a specific range of environments, and if the environment changes sufficiently then norms may become unsuitable. An emergent approach can cope with a changing environment, but by its very nature the organisation has little control over the norms that arise, and the learning process can be lengthy. Our approach allows the impact of norms designed offline to be monitored, so that poorly performing norms can be identified and replaced.

Centeno and Billhardt propose an adaptive normative system that tailors incentives to individual agents [6]. Their work provides a way to modify norms that are no longer useful because they are violated by agents since the incentives can be modified to ensure that agents will comply. However, they do not consider situations where compliance with a norm is itself detrimental to the organisation. Boella et al. do consider this situation [2] by using the concept of norm goals, where each norm has an associated intended goal, and provide a logical framework with which an agent can reason about

the desirability of compliance. However, this does not detect if the goal of the norm becomes detrimental, and raises the need for possibly sophisticated agent reasoning.

Regarding organisational performance, Centeno et al. provide a formal model of a normative organisation that includes the notion of a useful regulative system (or normative system) as one that improves organisational utility [7]. However, they do not quantify the effect of particular norms on organisational performance, instead considering the normative system as a whole, whereas our work specifically looks at individual norms. Dignum and Dignum investigate the fitness of an organisation's structure to its task [10] and propose a simulation approach to determine the effectiveness of that structure. We likewise use a simulation approach, although we focus upon the normative aspect of an organisation rather than its structure.

Norm monitoring within the literature has focused upon compliance issues (for example, [17]), rather than determining whether a norm is effective. With respect to changing norms to improve organisational performance, Koeppen et al. propose and implement a method to select efficient norms using case-based reasoning [14]. Their approach builds up a library of cases to select the best norm to improve simulated traffic flow in a multi-agent system, based on organisational goals.

7 Discussion and Further Work

We have presented a model that links the norms of an organisation to its task performance. The model considers the plan selection mechanism of the agents and how this selection is influenced by applicable norms. This in turn affects the actions taken by the agents in pursuit of their goals, and hence the changes in the environment that lead to the organisational goals being achieved, or not. We proposed a mechanism to enable an organisation to monitor the impact of norms upon performance over time, and implemented it using a scenario based upon a team of UAVs undertaking a search and rescue mission. Our experimental results showed that it is possible to quantitatively measure this impact, and to measure the change of impact as the environment changes.

Our scenario is simplistic, with a single norm and only limited reasoning by the agent about whether to comply or not. A more sophisticated implementation could include agents using local knowledge to determine whether it is valuable to violate a norm and hence investigate the tension between organisational norms intended to encourage long-term success and agent decisions geared toward short-term success.

In addition, in an organisation with multiple norms it may not be feasible to monitor the impact of every norm and possible subset of norms using our approach[4], so we need a method to decide which norms to monitor. Possible approaches include: assessing how sensitive a norm's impact is to environmental change, and then monitoring those most sensitive; grouping norms according to the roles and tasks to which they apply, and monitoring those applying to the most critical tasks.

Finally, it is important to note that the quality of the impact estimation is tied to the accuracy of the simulation. In a real world UAV scenario, it may be challenging to adequately model factors such as system failures and probability of victim detection.

[4] In a MAS with n norms, there are potentially 2^n combinations that could be monitored.

References

1. G. Andrighetto, M. Campenni, R. Conte, and M. Paolucci. On the immergence of norms: a normative agent architecture. In *Proc. of AAAI Symposium, Social and Organizational Aspects of Intelligence*, 2007.
2. G. Boella, G. Governatori, A. Rotolo, and L. Van Der Torre. Lex minus dixit quam voluit, lex magis dixit quam voluit: a formal study on legal compliance and interpretation. *AI Approaches to the Complexity of Legal Systems. Complex Systems, the Semantic Web, Ontologies, Argumentation, and Dialogue*, pages 162–183, 2010.
3. R. Bordini and J. Hübner. BDI agent programming in AgentSpeak using Jason. In F. Toni and P. Torroni, editors, *Computational Logic in Multi-Agent Systems*, volume 3900 of *LNCS*, pages 143–164. Springer, 2006.
4. M.E. Bratman, D.J. Israel, and M.E. Pollack. Plans and resource-bounded practical reasoning. *Computational intelligence*, 4(4):349–355, 1988.
5. K.M. Carley and L. Gasser. Computational organization theory. In G. Weiss, editor, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press.
6. R. Centeno and H. Billhardt. Using incentive mechanisms for an adaptive regulation of open multi-agent systems. In *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence*, pages 139–145, 2011.
7. R. Centeno, H. Billhardt, R. Hermoso, and S. Ossowski. Organising MAS: A formal model based on organisational mechanisms. In *Proc. of the 2009 ACM Symposium on Applied Computing*, pages 740–746. ACM, 2009.
8. M. Dastani, V. Dignum, and F. Dignum. Role-assignment in open agent societies. In *Proc. of the 2nd Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, pages 489–496. ACM, 2003.
9. F. Dignum, D. Morley, E.A. Sonenberg, and L. Cavedon. Towards socially sophisticated BDI agents. In *Proc. of the 4th Int. Conf. on MultiAgent Systems*, pages 111–118, 2000.
10. V. Dignum and F. Dignum. Understanding organizational congruence: formal model and simulation framework. In *Proc. of the 2007 Spring Simulation Multiconference - Volume 2*, pages 178–184, 2007.
11. P. Doherty and P. Rudol. A UAV search and rescue scenario with human body detection and geolocalization. In M.A. Orgun and J. Thornton, editors, *AI 2007: Advances in Artificial Intelligence*, volume 4830 of *LNCS*, pages 1–13. Springer, 2007.
12. M. Hannoun, O. Boissier, J. Sichman, and C. Sayettat. MOISE: An organizational model for multi-agent systems. In M. Monard and J. Sichman, editors, *Advances in Artificial Intelligence*, volume 1952 of *LNCS*, pages 156–165. Springer, 2000.
13. B. Horling and V. Lesser. A survey of multi-agent organizational paradigms. *The Knowledge Engineering Review*, 19(4):281–316, 2004.
14. J. Koeppen, M. Lopez-Sanchez, J. Morales, and M. Esteva. Learning from experience to generate new regulations. In *Proc. of the 6th Int. Conf. on Coordination, Organizations, Institutions, and Norms in Agent Systems*, pages 337–356. Springer, 2010.
15. F. Lopez y Lopez and M. Luck. Modelling norms for autonomous agents. In *Proc. of the 4th Mexican Int. Conf. on Computer Science*, pages 238–245, Sept. 2003.
16. K. A. Merchant. The control function of management. *Sloan Management Review*, 23(4):43–55, 1982.
17. S. Modgil, N. Faci, F. Meneguzzi, N. Oren, S. Miles, and M. Luck. A framework for monitoring agent-based normative systems. In *Proc. of the 8th Int. Conf. on Autonomous Agents and Multiagent Systems*, pages 153–160, 2009.
18. N. Oren, W. Vasconcelos, F. Meneguzzi, and M. Luck. Acting on norm constrained plans. In J. Leite, P. Torroni, T. Ågotnes, G. Boella, and L. Van Der Torre, editors, *Computational Logic in Multi-Agent Systems*, volume 6814 of *LNCS*, pages 347–363. Springer, 2011.

19. A. Rao. AgentSpeak(L): BDI agents speak out in a logical computable language. In W. Van de Velde and J. Perram, editors, *Agents Breaking Away*, volume 1038 of *LNCS*, pages 42–55. Springer, 1996.
20. S. Sen and S. Airiau. Emergence of norms through social learning. In *Proc. of the 20th Int. Joint Conf. on Artifical Intelligence*, pages 1507–1512, 2007.
21. Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: off-line design. *Artificial Intelligence*, 73(1–2):231–252, 1995.
22. R. Simons. Control in an age of empowerment. *Harvard Business Review*, 73(2):80–88, 1995.
23. W. Streeck and K. Thelen. Introduction: Institutional change in advanced political economies. In W. Streeck and K. Thelen, editors, *Beyond Continuity: Institutional Change in Advanced Political Economies*, pages 1–39. Oxford University Press, 2005.